The dawning of the autonomic computing era

by A. G. Ganek
T. A. Corbi

This issue of the *IBM Systems Journal* explores a broad set of ideas and approaches to autonomic computing—some first steps in what we see as a journey to create more self-managing computing systems. Autonomic computing represents a collection and integration of technologies that enable the creation of an information technology computing infrastructure for IBM's agenda for the next era of computing—e-business on demand. This paper presents an overview of IBM's autonomic computing initiative. It examines the genesis of autonomic computing, the industry and marketplace drivers, the fundamental characteristics of autonomic systems, a framework for how systems will evolve to become more self-managing, and the key role for open industry standards needed to support autonomic behavior in heterogeneous system environments. Technologies explored in each of the papers presented in this issue are introduced for the reader.

On March 8, 2001, Paul Horn, IBM Senior Vice President and Director of Research, presented the theme and importance of autonomic computing to the National Academy of Engineering at Harvard University. His message was:

> The information technology industry loves to prove the impossible possible. We obliterate barriers and set records with astonishing regularity. But now we face a problem springing from the very core of our success—and too few of us are focused on solving it. More than any other I/T problem, this one—if it remains unsolved—will actually prevent us from moving to the next era of computing. The obstacle is complexity . . . Dealing with it is the single most important challenge facing the I/T industry.[1]

One month later, Irving Wladawsky-Berger, Vice President of Strategy and Technology for the IBM Server Group, introduced the Server Group's autonomic computing project (then named eLiza*[2]), with the goal of providing self-managing systems to address those concerns. Thus began IBM's commitment to deliver "autonomic computing"—a new company-wide and, it is to be hoped, industry-wide, initiative targeted at coping with the rapidly growing complexity of operating, managing, and integrating computing systems.

We do not see a change in Moore's law[3] that would slow development as the main obstacle to further progress in the information technology (IT) industry. Rather, it is the IT industry's exploitation of the technologies in accordance with Moore's law that has led to the verge of a complexity crisis. Software developers have fully exploited a four- to six-orders-of-magnitude increase in computational power—producing ever more sophisticated software applications and environments. There has been exponential growth in the number and variety of systems and

©Copyright 2003 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

components. The value of database technology and the Internet has fueled significant growth in storage subsystems to hold petabytes[4] of structured and unstructured information. Networks have interconnected the distributed, heterogeneous systems of the IT industry. Our information society creates unpredictable and highly variable workloads on those networked systems. And today, those increasingly valuable, complex systems require more and more skilled IT professionals to install, configure, operate, tune, and maintain them.

IBM is using the phrase "autonomic computing"[5] to represent the vision of how IBM, the rest of the IT industry, academia, and the national laboratories can address this new challenge. By choosing the word "autonomic," IBM is making an analogy with the autonomic nervous system. The autonomic nervous system frees our conscious brain from the burden of having to deal with vital but lower-level functions. Autonomic computing will free system administrators from many of today's routine management and operational tasks. Corporations will be able to devote more of their IT skills toward fulfilling the needs of their core businesses, instead of having to spend an increasing amount of time dealing with the complexity of computing systems.

Need for autonomic computing

As Frederick P. Brooks, Jr., one of the architects of the IBM System/360*, observed, "Complexity is the business we are in, and complexity is what limits us."[6] The computer industry has spent decades creating systems of marvelous and ever-increasing complexity. But today, complexity itself is the problem.

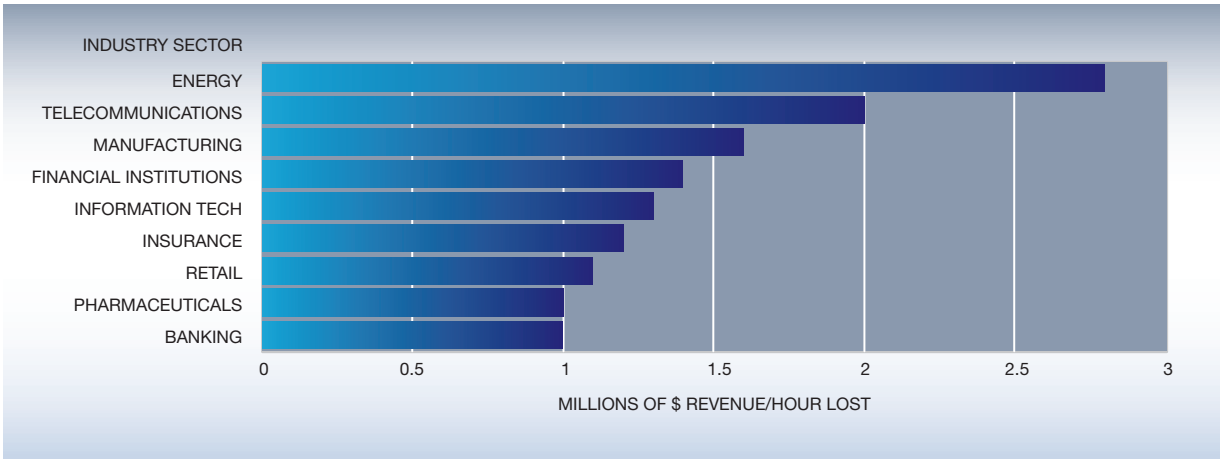
The spiraling cost of managing the increasing complexity of computing systems is becoming a significant inhibitor that threatens to undermine the future growth and societal benefits of information technology. Simply stated, managing complex systems has grown too costly and prone to error. Administering a myriad of system management details is too labor-intensive. People under such pressure make mistakes, increasing the potential of system outages with a concurrent impact on business. And, testing and tuning complex systems is becoming more difficult. Consider:

- It is now estimated that one-third to one-half of a company's total IT budget is spent preventing or recovering from crashes.[7]
- Nick Tabellion, CTO of Fujitsu Softek, said: "The commonly used number is: For every dollar to purchase storage, you spend $9 to have someone manage it."[8]
- Aberdeen Group studies show that administrative cost can account for 60 to 75 percent of the overall cost of database ownership (this includes administrative tools, installation, upgrade and deployment, training, administrator salaries, and service and support from database suppliers).[9]
- When you examine data on the root cause of computer system outages, you find that about 40 percent are caused by operator error,[10] and the reason is not because operators are not well-trained or do not have the right capabilities. Rather, it is because the complexities of today's computer systems are too difficult to understand, and IT operators and managers are under pressure to make decisions about problems in seconds.[11]
- A Yankee Group report[12] estimated that downtime caused by security incidents cost as much as $4,500,000 per hour for brokerages and $2,600,000 for banking firms.
- David J. Clancy, chief of the Computational Sciences Division at the NASA Ames Research Center, underscored the problem of the increasing systems complexity issues: "Forty percent of the group's software work is devoted to test," he said, and added, "As the range of behavior of a system grows, the test problem grows exponentially."[13]
- A recent Meta Group study looked at the impact of downtime by industry sector as shown in Figure 1.

Although estimated, cost data such as shown in Figure 1 are indicative of the economic impact of system failures and downtime. According to a recent IT resource survey by the Merit Project of Computer Associates International, 1867 respondents grouped the most common causes of outages into four areas of data center operations: systems, networks, database, and applications.[14] Most frequently cited outages included:

- For systems: operational error, user error, third-party software error, internally developed software problem, inadequate change control, lack of automated processes
- For networks: performance overload, peak load problems, insufficient bandwidth
- For database: out of disk space, log file full, performance overload
- For applications: application error, inadequate change control, operational error, nonautomated application exceptions

Figure 1 Downtime: Average hourly impact



Data from *IT Performance Engineering and Measurement Strategies: Quantifying Performance Loss*, Meta Group, Stamford, CT (October 2000).

Well-engineered autonomic functions targeted at improving and automating systems operations, installation, dependency management, and performance management can address many causes of these "most frequent" outages and reduce outages and downtime.

A confluence of marketplace forces are driving the industry toward autonomic computing. Complex heterogeneous infrastructures composed of dozens of applications, hundreds of system components, and thousands of tuning parameters are a reality. New business models depend on the IT infrastructure being available 24 hours a day, 7 days a week. In the face of an economic downturn, there is an increasing management focus on "return on investment" and operational cost controls—while staffing costs exceed the costs of technology. To compound matters further, there continues to be a scarcity of highly skilled IT professionals to install, configure, optimize, and maintain these complex, heterogeneous systems.

To respond, system design objectives must shift from the "pure" price/performance requirements to issues of robustness and manageability in the total-cost-of-ownership equation. As a profession, we must strive to simplify and automate the management of systems. Today's systems must evolve to become much more self-managing, that is: self-configuring, self-healing, self-optimizing, and self-protecting.

Irving Wladawsky-Berger outlined the solution at the Kennedy Consulting Summit in November 2001: "There is only one answer: The technology needs to manage itself. Now, I don't mean any far out AI project; what I mean is that we need to develop the right software, the right architecture, the right mechanisms . . . So that instead of the technology behaving in its usual pedantic way and requiring a human being to do everything for it, it starts behaving more like the 'intelligent' computer we all expect it to be, and starts taking care of its own needs. If it doesn't feel well, it does something. If someone is attacking it, the system recognizes it and deals with the attack. If it needs more computing power, it just goes and gets it, and it doesn't keep looking for human beings to step in."[15]

What is autonomic computing?

Automating the management of computing resources is not a new problem for computer scientists. For decades system components and software have been evolving to deal with the increased complexity of system control, resource sharing, and operational management. Autonomic computing is just the next logical evolution of these past trends to address the increasingly complex and distributed computing environments of today. So why then is this something new? Why a call to arms to the industry for heightened focus and new approaches? The answer lies in the radical changes in the information technology environment in just the few short years since the mid-1990s, with the use of the Internet and e-business extending environments to a dramatically

larger scale, broader reach, and a more mission-critical fundamental requirement for business. In that time the norm for a large on-line system has escalated from applications such as networks consisting of tens of thousands of fixed-function automated teller machines connected over private networks to rich suites of financial services applications that can be accessed via a wide range of devices (personal computer, notebook, handheld device, smart phone, smart card, etc.) by tens of millions of people worldwide over the Internet.

IBM's autonomic computing initiative has been outlined broadly. Paul Horn[1] described this "grand challenge" and called for industry-wide collaboration toward developing autonomic computing systems that have characteristics as follows:

- To be autonomic, a system needs to "know itself"—and consist of components that also possess a system identity.
- An autonomic system must configure and reconfigure itself under varying and unpredictable conditions.
- An autonomic system never settles for the status quo—it always looks for ways to optimize its workings.
- An autonomic system must perform something akin to healing—it must be able to recover from routine and extraordinary events that might cause some parts to malfunction.
- A virtual world is no less dangerous than the physical one, so an autonomic computing system must be an expert in self-protection.
- An autonomic computing system knows its environment and the context surrounding its activity, and acts accordingly.
- An autonomic system cannot exist in a hermetic environment (and must adhere to open standards).
- Perhaps most critical for the user, an autonomic computing system will anticipate the optimized resources needed to meet a user's information needs while keeping its complexity hidden.

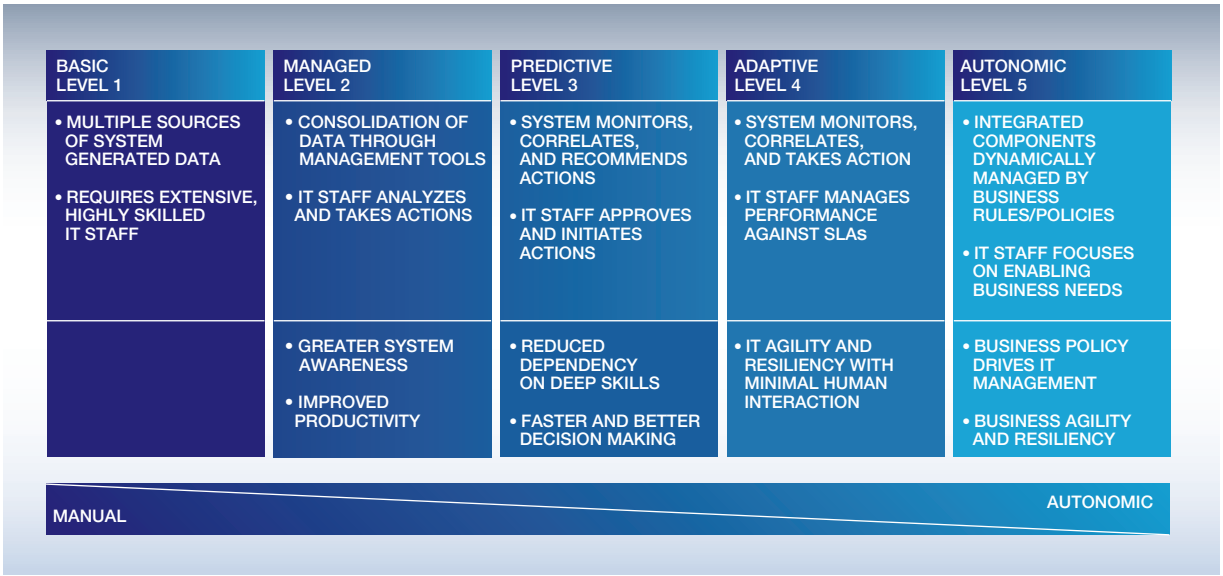
Fundamentals of autonomic computing. In order to incorporate these characteristics in "self-managing" systems, future autonomic computing systems will have four fundamental features. Various aspects of these four fundamental "self" properties are explored in this issue of the *IBM Systems Journal*.

Self-configuring. Systems adapt automatically to dynamically changing environments. When hardware and software systems have the ability to define themselves "on-the fly," they are self-configuring. This aspect of self-managing means that new features, software, and servers can be dynamically added to the enterprise infrastructure with no disruption of services. Systems must be designed to provide this aspect at a feature level with capabilities such as plug and play devices, configuration setup wizards, and wireless server management. These features will allow functions to be added dynamically to the enterprise infrastructure with minimum human intervention. Self-configuring not only includes the ability for each individual system to configure itself on the fly, but also for systems within the enterprise to configure themselves into the e-business infrastructure of the enterprise. The goal of autonomic computing is to provide self-configuration capabilities for the entire IT infrastructure, not just individual servers, software, and storage devices.

Self-healing. Systems discover, diagnose, and react to disruptions. For a system to be self-healing, it must be able to recover from a failed component by first detecting and isolating the failed component, taking it off line, fixing or isolating the failed component, and reintroducing the fixed or replacement component into service without any apparent application disruption. Systems will need to predict problems and take actions to prevent the failure from having an impact on applications. The self-healing objective must be to minimize all outages in order to keep enterprise applications up and available at all times. Developers of system components need to focus on maximizing the reliability and availability design of each hardware and software product toward continuous availability.

Self-optimizing. Systems monitor and tune resources automatically. Self-optimization requires hardware and software systems to efficiently maximize resource utilization to meet end-user needs without human intervention. IBM systems already include industry-leading technologies such as logical partitioning, dynamic workload management, and dynamic server clustering. These kinds of capabilities should be extended across multiple heterogeneous systems to provide a single collection of computing resources that could be managed by a "logical" workload manager across the enterprise. Resource allocation and workload management must allow dynamic redistribution of workloads to systems that have the necessary resources to meet workload requirements. Similarly, storage, databases, networks, and other resources must be continually tuned to enable efficient operations even in unpredictable environments. Fea-

Figure 2 Evolving to autonomic operations



From *IBM Global Services and Autonomic Computing*, IBM White Paper, October 2002; see http://www-3.ibm.com/autonomic/pdfs/wp-igs-autonomic.pdf.

tures must be introduced to allow the enterprise to optimize resource usage across the collection of systems within their infrastructure, while also maintaining their flexibility to meet the ever-changing needs of the enterprise.

Self-protecting. Systems anticipate, detect, identify, and protect themselves from attacks from anywhere. Self-protecting systems must have the ability to define and manage user access to all computing resources within the enterprise, to protect against unauthorized resource access, to detect intrusions and report and prevent these activities as they occur, and to provide backup and recovery capabilities that are as secure as the original resource management systems. Systems will need to build on top of a number of core security technologies already available today, including LDAP (Lightweight Directory Access Protocol), Kerberos, hardware encryption, and SSL (Secure Socket Layer). Capabilities must be provided to more easily understand and handle user identities in various contexts, removing the burden from administrators.

An evolution, not a revolution

To implement autonomic computing, the industry must take an evolutionary approach and deliver improvements to current systems that will provide significant self-managing value to customers without requiring them to completely replace their current IT environments. New open standards must be developed that will define the new mechanisms for interoperating heterogeneous systems. Figure 2 is a representation of those levels, starting from the basic level, through managed, predictive, and adaptive levels, and finally to the autonomic level.

As seen in the figure, the basic level represents the starting point where some IT systems are today. Each system element is managed independently by IT professionals who set it up, monitor it, and eventually replace it. At the managed level, systems management technologies can be used to collect information from disparate systems onto fewer consoles, reducing the time it takes for the administrator to collect and synthesize information as the systems become more complex to operate. In the predictive level, as new technologies are introduced that provide correlation among several elements of the system, the system itself can begin to recognize patterns, predict the optimal configuration, and provide advice on what course of action the administrator should take. As these technologies improve and as people become more comfortable with the advice and predictive power of these systems, we can pro-

Table 1 Aligning with business goals

Basic	Managed	Predictive	Adaptive	Autonomic
Process: Informal, reactive, manual	**Process:** Documented, improved over time, leverage of industry best practices; manual process to review IT performance	**Process:** Proactive, shorter approval cycle	**Process:** Automation of many resource mgmt best practices and transaction mgmt best practices, driven by service level agreements	**Process:** All IT service mgmt and IT resource mgmt best practices are automated
Tools: Local, platform- and product-specific	**Tools:** Consolidated resource mgmt consoles with problem mgmt system, automated software install, intrusion detection, load balancing	**Tools:** Role-based consoles with analysis and recommendations; product configuration advisors; real-time view of current & future IT performance; automation of some repetitive tasks; common knowledge base of inventory and dependency management	**Tools:** Policy management tools that drive dynamic change based on resource specific policies	**Tools:** Costing/financial analysis tools, business and IT modeling tools, trade-off analysis; automation of some e-business mgmt roles
Skills: Platform-specific, geographically dispersed with technology	**Skills:** Multiple skill levels with centralized triage to prioritize and assign problems to skilled IT professionals	**Skills:** Cross-platform system knowledge, IT workload management skills, some bus process knowledge	**Skills:** Service objectives and delivery per resource, and analysis of impact on business objectives	**Skills:** E-business cost & benefit analysis, performance modeling, advanced use of financial tools for IT context
Benchmarks: Time to fix problems and finish tasks	**Benchmarks:** System availability, time to close trouble tickets and work requests	**Benchmarks:** Business system availability, service level agreement attainment, customer satisfaction	**Benchmarks:** Business system response time, service level agreement attainment, customer satisfaction, IT contribution to business success	**Benchmarks:** Business success, competitiveness of service level agreement metrics, business responsiveness

Source: *Autonomic Computing Concepts*, IBM White Paper, October 2002; see: http://www-3.ibm.com/autonomic/pdfs/AC_Concepts.pdf.

gress to the adaptive level where the systems themselves can automatically take the correct actions based on the information that is available to them and the knowledge of what is happening in the systems. Service Level Agreements (SLAs)[16] guide operation of the system. Finally, at the fully autonomic level, the system operation is governed by business policies and objectives. Users interact with the system to monitor the business processes or alter the objectives.

As companies progress through the five levels of autonomic computing, the processes, tools, and benchmarks become increasingly sophisticated, and the skills requirement becomes more closely aligned with the business. Table 1 illustrates this correlation.

The basic level represents the starting point for most IT organizations. If they are formally measured at all, they are typically measured on the time required to finish major tasks and fix major problems. The IT organization is viewed as a cost center, in which the variable costs associated with labor are preferred over an investment in centrally coordinated systems management tools and processes.

At the managed level, IT organizations are measured on the availability of their managed resources, their

time to close trouble tickets in their problem management system, and their time to complete formally tracked work requests. To improve on these measurements, IT organizations document their processes and continually improve them through manual feedback loops and adoption of best practices. IT organizations gain efficiency through consolidation of management tools to a set of strategic platforms and through a hierarchical problem management triage organization.

In the predictive level, IT organizations are measured on the availability and performance of their business systems and their return on investment. To improve on these measurements, IT organizations measure, manage, and analyze transaction performance. The implications of the critical nature of the role of the IT organization in the success of the business are understood. Predictive tools are used to project future IT performance, and many tools make recommendations to improve future performance.

In the adaptive level, IT resources are automatically provisioned and tuned to optimize transaction performance. Business policies, business priorities, and service-level agreements guide the autonomic infrastructure behavior. IT organizations are measured on end-to-end business system response times (transaction performance), the degree of efficiency with which the IT infrastructure is utilized, and their ability to adapt to shifting workloads.

In the autonomic level, IT organizations are measured on their ability to make the business successful. To improve business measurements, IT tools understand the financial metrics associated with e-business activities and supporting IT activities. Advanced modeling techniques are used to optimize e-business performance and quickly deploy newly optimized e-business solutions.

Today's software and hardware system components will evolve toward more autonomic behavior. For example:

- Data management. New database software tools can use statistics from the databases, analyze them, and learn from the historical system performance information. The tools can help an enhanced database system automatically detect potential bottlenecks as they are about to occur and attempt to compensate for them by adjusting tuning parameters. Query optimizers can learn the optimal index and route to certain data and automatically seek out that path based on the historical access patterns and associated response times.

- Web servers and software. Web servers can provide real-time diagnostic "dashboard" information, enabling customers to more quickly become aware of resource problems, instead of relying on after-the-fact reports to identify problems. Once improved instrumentation is available, autonomic functions can be introduced that enable the Web server infrastructure to automatically monitor, analyze, and fix performance problems. As an example, suppose an application server is freezing-up intermittently, and no customer transactions are being processed for several seconds, thus losing thousands of dollars in business, as well as customer confidence and loyalty. Using real-time monitoring, predictive analysis, and auto-tuning, the freeze-up is anticipated before it happens. The autonomic function compares real-time data with historical problem data (i.e., suggesting that the cache sizes were set too low). The settings are reset automatically without service disruption, and a report is sent to the administrator that shows what action was taken.

- Systems management. Systems management software can contain improved problem determination and data collection features designed to help businesses better diagnose and prevent interruptions (or breaches of security). Such systems management software must enable customers to take an "end-to-end" view of their computing environment across multiple, independently installed hardware and software elements. A bank transaction, for example, might "touch" a discrete database, another transaction, and Web application servers as it is processed across a network. If a problem occurs with processing on one of the individual components, lack of an integrated problem determination infrastructure makes it more difficult to determine what prevented that bank transaction from completing successfully. A consolidated view created by the system management software would enable the system and IT staffs to identify and quickly react to problems as they happen by providing an end-to-end view of the application. The end-to-end view of the environment allows companies to understand problems and performance information in the context of their business goals.

- Servers. Computers can be built that need less human supervision. Computers can try to fix themselves in the event of a failure, protect themselves

from hacker attacks, and configure themselves when adding new features. Servers can use software algorithms that learn patterns in Internet traffic or application usage, and provision resources in a way that gives the shortest response time to the task with the highest business priority. Server support for heterogeneous and enterprise workload management, dynamic clustering, dynamic partitioning, improved setup wizards, improved user authentication, directory integration, and other tools to protect access to network resources are all steps toward more autonomic functioning.

IBM hardware and software systems have already made significant progress in introducing autonomic computing functionality.[2] But there is much more work ahead. The efforts to achieve cohesive system behavior must go beyond improvements in the individual components alone. These components must be federated, employing an integrating architecture that establishes the instrumentation, policy, and collaboration technologies so that groups of resources can work in concert, as for example, across systems in a grid. System management tools will play a central role in coordinating the actions of system components, providing a simplified mechanism for system administration and for translating business objectives into executable policies to govern the actions of the IT resources available.

Industry standards are needed to support autonomic computing

Most IT infrastructures are composed of components supplied by different vendors. Open industry standards are the key to the construction of autonomic computing systems. Systems will need more standardization to introduce a uniform approach to instrumentation and data collection, dynamic configuration, and operation. Uniformity will allow the intersystem exchange of instrumentation and control information to create the basis for collaboration and autonomic behavior among heterogeneous systems.

For example, in storage systems, a standard that has been proposed for specifying data collection items is the Bluefin specification. Bluefin[17] defines a language and schema that allow users to reliably identify, classify, monitor, and control the physical and logical devices in storage area networking. The Storage Networking Industry Association (SNIA) has taken this standard to the Distributed Management Task Force (DMTF). SNIA is using Bluefin as the basis for its storage management initiative, the intent of which is to become the SNIA standard for management.

In the case of application instrumentation, the standard that has been proposed for obtaining the transaction rate, response time, failure rate, and topology data from applications is the Open Group Application Response Measurement (ARM)[18] application programming interfaces (APIs). The Application Response Measurement API defines the function calls that can be used to instrument an application or other software for transaction monitoring. It provides a way to monitor business transactions by embedding simple cells in the software that can be captured by an agent supporting the ARM API. The calls are used to capture data, allowing software to be monitored for availability, service levels, and capacity.

Other standards, such as the DMTF Common Information Model (CIM)[19] and Web Service Level Agreement (WSLA), provide languages and schemas for defining the available data. CIM is an object-oriented information model that provides a conceptual view of physical and logical system components. WSLA is a language to express SLA contracts, to support guaranteed performance, and to handle complex dynamic fluctuations in service demand. SLA-based system management would enable service providers to offer the same Web service at different performance levels, depending on contracts with their customers. WSLA is available through the IBM alphaWorks* Web Services Toolkit[20] that features a WSLA document approach based on Extensible Markup Language (XML) to define SLAs.

These standards are technologies that enable the building of "inter-communicating" autonomic system elements that are the foundation for cooperation in a federation of system components. Each individual autonomic "element" is responsible for managing itself, that is, for configuring itself internally, for healing internal failures when possible, for optimizing its own behavior, and for protecting itself from external probing and attack.

Autonomic elements are the building blocks for making autonomic systems. Autonomic elements continuously monitor system (or component) behavior through "sensors" and make adjustments through "effectors." By monitoring behavior through sensors, analyzing those data, then planning what action should be taken next (if any), and executing that ac-

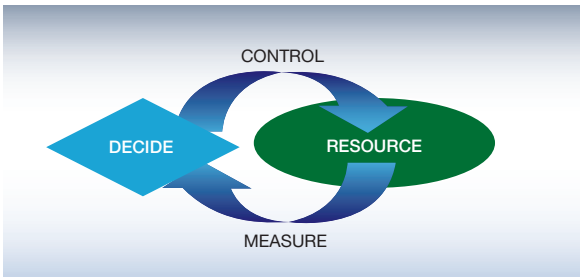
tion through effectors, a kind of "control loop"[21] is created (see Figure 3).

Interconnecting autonomic elements requires distributed computing mechanisms to access resources across the network. "Grid computing"[22] encompasses the idea of an emerging infrastructure that is focused on networking together heterogeneous, multiple regional and national computing systems. It has been called the next evolutionary step for the Internet. The term "grid" was chosen as an analogy with the electric power grid, which supplies pervasive access to power. Grids are persistent computing environments that enable software applications to integrate instruments, displays, and computational and information resources that are managed by diverse organizations in widespread locations.

In 2001, the Globus Project[23,24] launched a research and development program aimed at creating a toolkit based on the Open Grid Service Architecture (OGSA) that defines standard mechanisms for creating, naming, and discovering services and specifies various protocols to support accessing services. Essentially, OGSA is a framework for distributed computing, based on Web services protocols. Although OGSA is a proposed standard that will be developed and defined in the Global Grid Forum (GGF),[25] it is applicable whether the environment consists of a multiorganization grid or simply distributed resources within an enterprise. IBM, Microsoft Corporation, and others have already announced support for the OGSA framework. Work efforts on grid and OGSA are creating important architectural models and new open industry standards that are enablers for the IT industry to make progress toward more self-managing systems. Since grid deployments can expand the domain of computing across many systems, in our view, a successful grid system will require autonomic functionality.

Individual autonomic elements can interact through OGSA mechanisms. For example, today there is no accepted "sensor and effector" standard. But, the Globus Toolkit provides information services utilities to provide information about the status of grid resources. One of these utilities is the Monitoring and Discovery Service (MDS),[26] MDS 2.2 GRIS "Information Providers" that are essentially sensors or probes. The Globus Toolkit also provides a mechanism for authenticated access to MDS. Fault detection allows a client process to be monitored by a heartbeat monitor. Resource management APIs provide some job management capabilities.

Figure 3 Control loop



From *Autonomic Computing Concepts*, IBM White Paper, 2001

Thus we are seeing the emergence of some basic standard mechanisms needed for distributed "control loops" that in turn are needed for autonomic computing. When control loop standards are in place, the industry must address the more complex issues of specifying and automating policy management and service level agreements (SLAs).

A typical enterprise has a heterogeneous set of routers, firewalls, Web servers, databases, and workstations, all with different system management mechanisms. So again, industry standards will be needed in order to enable true policy management. We expect that policy specifications will be widely used in enterprises for defining quality of service management, storage backup, and system configuration, as well as security authorization and management.

A common approach to specifying and deploying policy would enable an enterprise to define and disseminate policies that reflect its overall IT service goals. A common, standard set of tools and techniques used throughout the enterprise could simplify analysis and reduce inconsistencies and conflicts in the policies deployed across the various components within the enterprise and also allow a policy exchange with external service providers.

Various standards bodies are working on specifying policies for network and systems management, security, and role-based access control (RBAC). The Internet Engineering Task Force (IETF)[27] and DMTF[28] have been concentrating on information models for management policies, protocols for transferring policies to network devices, and routing policies; the National Institute of Standards and Technology (NIST)[29] is working toward an RBAC standard; and the Oasis consortium (Organization for the Advancement of

Structured Information Standards)[30] is working on an XML-based specification of access control policies and authentication information.

It will take some time for the current divergent standards policy-based solutions to come to embrace a common approach. Meanwhile, research on policy-based management approaches continues.[31,32] Advances in policy management are needed to enable enterprises to eventually specify the behaviors of IT services in terms of the business process objectives of the enterprises.

Exploratory research and development presented in this issue

Autonomic computing represents an exciting new research direction in computing. IBM believes that meeting the grand challenge of autonomic computing systems will involve researchers in a diverse array of fields, including systems management, distributed computing, networking, operations research, software development, storage, artificial intelligence, and control theory, as well as others.

The challenge of autonomic computing requires more than the re-engineering of today's systems. Autonomic computing also requires new ideas, new insights, and new approaches. This issue of the *IBM Systems Journal* provides just a glimpse into an array of research and development efforts underway for autonomic computing. Below we present the topics in the issue.

D. C. Verma, S. Sahu, S. Calo, A. Shaikh, I. Chang, and A. Acharya in their paper, "SRIRAM: A Scalable Resilient Autonomic Mesh,"[33] propose a method that facilitates instantiating mirroring and replication of services in a network of servers.

The ability to redistribute hardware resources dynamically is essential to both the self-configuring and self-optimizing goals of autonomic computing. J. Jann, L. M. Browning, and R. S. Burugula describe this new server capability in "Dynamic Reconfiguration: Basic Building Blocks for Autonomic Computing on IBM pSeries Servers."[34]

In the first of two invited papers, D. A. Norman and A. Ortony from Northwestern University, along with D. M. Russell of IBM, discuss in "Affect and Machine Design: Lessons for the Development of Autonomous Machines"[35] how studying the human characteristics of cognition and affect will help designers in developing complex autonomic systems that will interact with unpredictable situations.

K. Whisnant, Z. T. Kalbarczyk, and R. K. Iyer examine the difficulties of dynamically reconfiguring application software in their paper, "A System Model for Dynamically Reconfigurable Software."[36] They believe that both static structure and run-time behaviors must be captured in order to define a workable reconfiguration model.

One technology to support self-healing and self-configuring is the ability to dynamically insert new pieces of software and remove other pieces of code, without shutting down the running system. This technology is being explored in the K42 research operating system and is presented in the paper by J. Appavoo, K. Hui, C. A. N. Soules, R. W. Wisniewski, D. M. Da Silva, O. Krieger, M. A. Auslander, D. J. Edelsohn, B. Gamsa, G. R. Ganger, P. McKenney, M. Ostrowski, B. Rosenburg, M. Stumm, and J. Xenidis, entitled "Enabling Autonomic Behavior in Systems Software with Hot Swapping."[37]

L. W. Russell, S. P. Morgan, and E. G. Chron introduce the idea of a predictive autonomic system in their paper entitled "Clockwork: A New Movement in Autonomic Systems."[38] They explore the idea of a system that anticipates workload needs based on statistical modeling, tracking, and forecasting.

Component-based development, where multiple distributed software components are composed to deliver a particular business function, is an emerging programming model in the Web services world. D. M. Yellin in his paper, "Competitive Algorithms for the Dynamic Selection of Component Implementations,"[39] proposes a strategy and framework for optimizing component performance based on switching between different component implementations.

In an example of "self-optimizing," V. Markl, G. M. Lohman, and V. Raman discuss improving query performance by comparing estimates with actual results toward self-validating query planning in "LEO: An Autonomic Query Optimizer for DB2."[40]

As noted, system and network security are fundamental to autonomic computing systems. In "Security in an Autonomic Computing Environment,"[41] D. M. Chess, C. C. Palmer, and S. R. White outline a number of security and privacy issues in the design and development of autonomic systems.

G. Lanfranchi, P. Della Peruta, A. Perrone, and D. Calvanese describe what they see as a paradigm shift in system management needed for autonomic computing. In their paper, "Toward a New Landscape of Systems Management in an Autonomic Computing Environment,"[42] they introduce a knowledge-based resource model technology that extends across design, delivery, and run time.

In the second invited paper, "Comparing Autonomic and Proactive Computing,"[43] R. Want, T. Pering, and D. Tennenhouse of Intel Research present a high-level discussion of the similarities between proactive computing and autonomic computing with an emphasis on their research in proactive computing—an environment in which computers anticipate what users need and act accordingly.

Today, optimizing performance in multisystem e-commerce environments requires considerable skill and experience. In "Managing Web Server Performance with AutoTune Agents,"[44] Y. Diao, J. L. Hellerstein, S. Parekh, and J. P. Bigus describe intelligent agents that use control theory techniques to autonomically adjust an Apache** Web server to dynamic workloads.

The backbone of a grid or typical autonomic computing system will be an intelligent, heterogeneous network infrastructure. Management issues related to topology, service placement, cost and service metrics, as well as dynamic administration structure are explored by R. Haas, P. Droz, and B. Stiller in "Autonomic Service Deployment in Networks."[45]

Although much of the discussion on autonomic computing often focuses on servers, networks, databases, and storage management, we realize that personal computer users would also benefit greatly by the introduction of autonomic features. D. F. Bantz, C. Bisdikian, D. Challener, J. P. Karidis, S. Mastrianni, A. Mohindra, D. G. Shea, and M. Vanover explore these possibilities in their paper, "Autonomic Personal Computing."[46]

People will still need to interact with autonomic computing systems. D. M. Russell, P. P. Maglio, R. Dordick, and C. Neti in their paper entitled "Dealing with Ghosts: Managing the User Experience of Autonomic Computing"[47] argue that the lessons we have learned in human-computer interaction research must be applied to effectively expose and communicate the run-time behavior of these complex systems and to better define and structure the user system operation scenarios.

In the Technical Forum section, the complex challenges of life-cycle management and providing capacity on demand are examined in a project as described by A. Abbondanzio, Y. Aridor, O. Biran, L. L. Fong, G. S. Goldszmidt, R. E. Harper, S. M. Krishnakumar, G. Pruett, and B.-A. Yassur in "Management of Application Complexes in Multitier Clustered Systems."[48]

Conclusion

In his keynote speech at the Almaden Institute 2002,[49] John Hennessy, President of Stanford University, presented his view of the autonomic challenge. While acknowledging the significant accomplishments in hardware architecture over the past 20 years, he urged industry and academia to look forward and to shift focus to a set of issues related to how services will be delivered over networks in the Internet/Web-centric "post-desktop" era: "As the business use of this environment grows and as people become more and more used to it, the flakiness that we've all accepted in the first generation of the Internet and the Web—will become unacceptable." Hennessy emphasized an increased research focus on availability, maintainability, scalability, cost, and performance—all fundamental aspects of autonomic computing.

Autonomic computing is a journey. Progress will be made in a series of evolutionary steps. This issue of the *IBM Systems Journal* presents some of the technology signposts that can serve to guide the ongoing research in this new direction.

Acknowledgments

First, we express our thanks to the many authors who submitted their work for publication in this issue. Special thanks go to Lorraine Herger, Kazuo Iwano, Pratap Pattnaik, Connie Marconi, and Felicia C. Medley, who organized the call for papers, managed the process to select the topics and the papers, and worked with the authors and *IBM Systems Journal* staff to produce this issue. We also thank the numerous referees whose comments helped all the authors in improving their papers.

We particularly express our appreciation to Paul Horn, IBM Senior Vice President and Director of Research, for launching the autonomic computing

grand challenge in his 2001 presentation on autonomic computing and to Irving Wladawsky-Berger who led the launch of the autonomic computing project in the IBM Server Group.

We acknowledge the efforts of Robert Morris, Anant Jhingran, Tushar Chandra, and K. M. Moiduddin, who organized the Almaden Autonomic Computing Institute held at San Jose, California in April 2002. We also acknowledge Sam S. Adams, Lisa F. Spainhower, William H. Tetzlaff, William H. Chung, Steve R. White, and Kazuo Iwano who organized the Autonomic Computing Conference sponsored by the IBM Academy of Technology and held in Yorktown Heights, New York in May 2002. Our thanks go to Christopher W. Luongo for his article, "Server to IT: Thanks, But I Fixed It Myself."

The authors would like to thank Ric Telford, John Sweitzer, and Jim Crosskey of the Autonomic Computing department for their contributions to this manuscript. The authors also wish to acknowledge the *IBM Systems Journal* staff for their many helpful suggestions during the creation of this issue.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Apache Digital Corporation.

Cited references and notes

1. P. Horn, *Autonomic Computing: IBM's Perspective on the State of Information Technology*, IBM Corporation (October 15, 2001); available at http://www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf.
2. IBM Server Group, *eLiza: Building an Intelligent Infrastructure for E-business—Technology for a Self-Managing Server Environment*, G520-9592-00, IBM Corporation (2001); also at http://www-1.ibm.com/servers/eserver/introducing/eliza/eliza_final.pdf.
3. For more than 30 years, Moore's Law has forecasted progress in the computing industry like an immutable force of nature. In 1965, Gordon E. Moore, then the director of research and development at Fairchild Semiconductor Corporation, made the casual observation that processing power will double every 18 to 24 months, suggesting healthy growth ahead over the next decade for the then-nascent silicon chip industry. Five years later, Moore cofounded Intel Corporation, and "Moore's law" was well on its way to becoming a self-fulfilling prophecy among researchers and developers in the industry.
4. A petabyte (PB) is 1000 terabytes, and a terabyte is 1000 gigabytes. CERN, the European Center for Nuclear Research located just outside of Geneva, Switzerland, has started building a 100-PB archive to house data from the particle accelerator of the research center. See "From Kilobytes to Petabytes in 50 Years," *Science and Technology Review*, UCRL-52000-02-4, Lawrence Livermore National Laboratory, Livermore, CA (April 15, 2002), http://www.llnl.gov/str/March02/March50th.html.
5. The term "autonomic computing" comes from an analogy to the autonomic central nervous system in the human body, which adjusts to many situations without any external help. "Autonomic" is defined as: (a) Of, relating to, or controlled by the autonomic nervous system; (b) Acting, or occurring involuntarily; automatic; an autonomic reflex.
6. F. P. Brooks, Jr., *The Mythical Man-Month: Essays on Software Engineering*, Twentieth Anniversary Edition, Addison-Wesley Publishing Co., Reading, MA (1995), p. 226. See also, F. P. Brooks, Jr., "No Silver Bullet: Essence and Accidents of Software Engineering," *Computer* **20**, No. 4, 10–19 (1987).
7. D. A. Patterson, A. Brown, P. Broadwell, G. Candea, M. Chen, J. Cutler, P. Enriquez, A. Fox, E. Kiciman, M. Merzbacher, D. Oppenheimer, N. Sastry, W. Tetzlaff, J. Traupman, N. Treuhaft, *Recovery-Oriented Computing (ROC): Motivation, Definition, Techniques, and Case Studies*, U.C. Berkeley Computer Science Technical Report, UCB//CSD-02-1175, University of California, Berkeley (March 15, 2002).
8. K. Evans-Correia, "Simplifying Storage Management Starts with More Efficient System Utilization," Interview with N. Tabellion, *searchStorage* (August 29, 2001), see http://searchstorage.techtarget.com/qna/0,289202,sid5_gci764063,00.html.
9. *IBM Data Management Tools: New Opportunities for Cost-Effective Administration*, Profile Report, Aberdeen Group, Inc., Boston (April 2002), p. 3.
10. D. Patterson, "Availability and Maintainability >> Performance: New Focus for a New Century," *USENIX Conference on File and Storage Technologies* (FAST '02), Keynote Address, Monterey, CA (January 29, 2002).
11. A. Brown and D. A. Patterson, "To Err Is Human," *Proceedings of the First Workshop on Evaluating and Architecting System dependabilitY* (EASY '01), Goeteborg, Sweden (July 2001).
12. "How Much Is an Hour of Downtime Worth to You?" from *Must-Know Business Continuity Strategies*, Yankee Group, Boston (July 31, 2002).
13. D. J. Clancy, "NASA Challenges in Autonomic Computing," *Almaden Institute 2002*, IBM Almaden Research Center, San Jose, CA (April 10, 2002).
14. Merit Project, Computer Associates International, http://www.meritproject.com/it_survey_results.htm.
15. I. Wladawsky-Berger, "Advancing E-business into the Future: The Grid," *Kennedy Consulting Summit 2001*, New York (November 29, 2001).
16. In this context, a Service Level Agreement (SLA) is a compact between a customer or consumer and a provider of an IT service that specifies the levels of availability, serviceability, performance (and tracking/reporting), problem management, security, operation, or other attributes of the service, often established via negotiation. Typically, an SLA identifies and defines the customer's needs, provides a framework for discussion and understanding, attempts to simplify complex requirements, outlines methods for resolving disputes, and helps eliminate unrealistic expectations.
17. "'Bluefin' A Common Interface for SAN Management," White Paper, Storage Networking Industry Association (August 13, 2002), http://www.snia.org/tech_activities/SMI/bluefin/Bluefin_White_Paper_v081302.pdf from http://www.snia.org/tech_activities/SMI/bluefin/.
18. *Application Response Measurement Issue 3.0 Java Binding*, Open Group Technical Standard CO14, The Open Group (October 2001), at http://www.opengroup.org/products/publications/catalog/c014.htm.

19. *Common Information Model (CIM) Specification Version 2.2*, DSP0004, Distributed Management Task Force (June 14, 1999), at http://www.dmtf.org/standards/standard_cim.php.
20. Web Services Toolkit, alphaWorks, IBM Corporation (July 26, 2000), http://www.alphaworks.ibm.com/tech/webservices toolkit.
21. The control loop is the essence of automation. By measuring or sensing some activity in a process to be controlled, a controller component decides what needs to be done next and executes the required operations through a set of actuators. The controller then remeasures the process to determine whether the actions of the actuator had the desired effect. The whole routine is then repeated in a continuous loop of measure, decide, actuate, and repeat.
22. *The Grid: Blueprint for a New Computing Infrastructure*, I. Foster and C. Kesselman, Editors, Morgan Kaufmann Publishers, Inc., San Francisco, CA (1999).
23. See http://www.globus.org, the home of the Globus Project, the Globus Toolkit (GT3), and work related to the Open Grid Services Architecture (OGSA).
24. I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of High Performance Computing* **15**, No. 3, 200–222 (2001); see also, http://www.globus.org/research/papers/anatomy.pdf.
25. Global Grid Forum, http://www.gridforum.org/.
26. *MDS 2.2 User's Guide*, The Globus Project, available at www.globus.org/mds/mdsusersguide.pdf.
27. Internet Engineering Task Force, http://www.dmtf.org.
28. Distributed Management Task Force, http://www.dmtf.org.
29. National Institute of Standards and Technology, http://www.nist.org.
30. Organization for the Advancement of Structured Information Standards, http://www.oasis-open.org.
31. L. Lymberopoulos, E. Lupu, and M. Sloman, "An Adaptive Policy Based Management Framework for Differentiated Services Networks," *Proceedings of the 3rd IEEE Workshop on Policies for Distributed Systems and Networks (Policy 2002)*, Monterey, CA (June 2002), pp. 147–158.
32. V. Sander, W. A. Adamson, I. Foster, and A. Roy, "End-to-End Provision of Policy Information for Network QoS," *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing* (HPDC-10), IEEE Press (August 2001).
33. D. C. Verma, S. Sahu, S. Calo, A. Shaikh, I. Chang, and A. Acharya, "SRIRAM: A Scalable Resilient Autonomic Mesh," *IBM Systems Journal* **42**, No. 1, 19–28 (2003, this issue).
34. J. Jann, L. A. Browning, and R. S. Burugula, "Dynamic Reconfiguration: Basic Building Blocks for Autonomic Computing on IBM pSeries Servers," *IBM Systems Journal* **42**, No. 1, 29–37 (2003, this issue).
35. D. A. Norman, A. Ortony, and D. M. Russell, "Affect and Machine Design: Lessons for the Development of Autonomous Machines," *IBM Systems Journal* **42**, No. 1, 38–44 (2003, this issue).
36. K. Whisnant, Z. T. Kalbarczyk, and R. K. Iyer, "A System Model for Dynamically Reconfigurable Software," *IBM Systems Journal* **42**, No. 1, 45–59 (2003, this issue).
37. J. Appavoo, K. Hui, C. A. N. Soules, R. W. Wisniewski, D. M. Da Silva, O. Krieger, M. A. Auslander, D. J. Edelsohn, B. Gamsa, G. R. Ganger, P. McKenney, M. Ostrowski, B. Rosenburg, M. Stumm, and J. Xenidis, "Enabling Autonomic Behavior in Systems Software with Hot Swapping," *IBM Systems Journal* **42**, No. 1, 60–76 (2003, this issue).
38. L. W. Russell, S. P. Morgan, and E. G. Chron, "Clockwork: A New Movement in Autonomic Systems," *IBM Systems Journal* **42**, No. 1, 77–84 (2003, this issue).
39. D. M. Yellin, "Competitive Algorithms for the Dynamic Selection of Component Implementations," *IBM Systems Journal* **42**, No. 1, 85–97 (2003, this issue).
40. V. Markl, G. M. Lohman, and V. Raman, "LEO: An Autonomic Query Optimizer for DB2," *IBM Systems Journal* **42**, No. 1, 98–106 (2003, this issue).
41. D. M. Chess, C. C. Palmer, and S. R. White, "Security in an Autonomic Computing Environment," *IBM Systems Journal* **42**, No. 1, 107–118 (2003, this issue).
42. G. Lanfranchi, P. Della Peruta, A. Perrone, and D. Calvanese, "Toward a New Landscape of Systems Management in an Autonomic Computing Environment," *IBM Systems Journal* **42**, No. 1, 119–128 (2003, this issue).
43. R. Want, T. Pering, and D. Tennenhouse, "Comparing Autonomic and Proactive Computing," *IBM Systems Journal* **42**, No. 1, 129–135 (2003, this issue).
44. Y. Diao, J. L. Hellerstein, S. Parekh, and J. P. Bigus, "Managing Web Server Performance with AutoTune Agents," *IBM Systems Journal* **42**, No. 1, 136–149 (2003, this issue).
45. R. Haas, P. Droz, and B. Stiller, "Autonomic Service Deployment in Networks," *IBM Systems Journal* **42**, No. 1, 150–164 (2003, this issue).
46. D. F. Bantz, C. Bisdikian, C. Challener, J. P. Karidis, S. Mastrianni, A. Mohindra, D. G. Shea, and M. Vanover, "Autonomic Personal Computing," *IBM Systems Journal* **42**, No. 1, 165–176 (2003, this issue).
47. D. M. Russell, P. P. Maglio, R. Dordick, and C. Neti, "Dealing with Ghosts: Managing the User Experience of Autonomic Computing," *IBM Systems Journal* **42**, No. 1, 177–188 (2003, this issue).
48. A. Abbondanzio, Y. Aridor, O. Biran, L. L. Fong, G. S. Goldszmidt, R. E. Harper, S. M. Krishnakumar, G. Pruett, and B.-A. Yassur, "Management of Application Complexes in Multitier Clustered Systems," Technical Forum, *IBM Systems Journal* **42**, No. 1, 189–195 (2003, this issue).
49. J. Hennessy, "Back to the Future: Time to Return to Some Long-Standing Problems in Computer Science," *Almaden Institute 2002*, IBM Almaden Research Center, San Jose, CA (April 10, 2002).

Accepted for publication November 1, 2002.

Alan G. Ganek *IBM Software Group, Hawthorne II, 17 Sky Line Drive, Hawthorne, New York 10532 (electronic mail: ganek@us.ibm.com).* Mr. Ganek is Vice President, Autonomic Computing, IBM Software Group. In this recently created role, he leads the IBM corporate-wide initiative for autonomic computing that focuses on making computing systems more self-managing and resilient, lowering the cost of ownership and removing obstacles to growth and flexibility. Prior to joining the Software Group, he was responsible for the technical strategy and operations of the IBM Research Division. Mr. Ganek joined IBM as a software engineer in 1978 in Poughkeepsie, New York, where he was involved in operating system design and development, computer addressing architecture, and parallel systems architecture and design. He was the recipient of IBM Outstanding Innovation Awards for his work on Enterprise Systems Architecture/370™ and System/390® Parallel Sysplex® Design. In 1985, he was appointed manager of MVS™ Design and Performance Analysis and was responsible for the technical plan and content of the MVS control program. Subsequently he was appointed VM/XA advanced system manager, responsible for strat-

egy, design, planning, customer support, system evaluation, and product delivery and control. Mr. Ganek became manager of Enterprise Systems Market Operations in 1989, responsible for System/390 software requirements and announcement strategy. He was appointed Director of Worldwide Software Manufacturing Strategy in 1990, responsible for IBM's strategy for manufacturing, distribution, and packaging of software, software service, and publications across all hardware platforms. In 1992, he was named Programming Systems Director, Quality and Development Operations, leading quality programs for the Programming Systems Division. He joined the Telecommunications and Media Industry Unit in 1994 as Director of Solutions Development, IBM Telecommunications and Media Industry Unit. Mr. Ganek received his M.S. in computer science from Rutgers University in 1981. He holds 15 patents.

Thomas A. Corbi *IBM Software Group, Hawthorne II, 17 Sky Line Drive, Hawthorne, New York 10532 (electronic mail: groklib@us.ibm.com).* Mr. Corbi is a member of the Autonomic Computing Architecture team that is working toward creating a framework and standards to enable development of autonomic software systems. He also manages the Autonomic Computing Technology Institute, which is a joint research program between the Software Group and the IBM Research Division. Prior to joining the Software Group, he was a research staff member and technical assistant to the Vice President of Emerging Business Opportunities in IBM Research. He joined IBM at the former Palo Alto Development Center as a programmer in 1974 after graduating from Yale University with a degree in computer science. He worked in the former General Products Division on future systems text-processing, data management, and database systems, and was development manager for IMS™/VS Fast Path at the IBM Santa Teresa Laboratory. Mr. Corbi joined the Research Division at the Thomas J. Watson Research Center in 1982 and managed the Program Understanding project from 1986 to 1988. He cochaired the 1988 IEEE Conference on Software Maintenance (CSM-88). From 1988 to 1990, he went on assignment as technical assistant to the S/390 Operating Systems Division Director in Poughkeepsie, New York, where he wrote "Understanding and Improving Software Development Productivity." He returned to Research and was senior manager of speech recognition development in the Experimental Software Development Center. In 1994, he was assigned to the Research Headquarters Technical Staff, assisting the Vice President for Technical Plans and Controls. Afterwards, Mr. Corbi contributed to a number of research projects, including low power exploratory systems, home networking and consumer systems, pervasive computing systems solutions, and wearable computing platforms. He holds two patents.